KAPITOLA 5

Práce s textovými řetězci

V této kapitole najdete:

- Deklarace textových řetězců
- Operace s řetězci
- Ukázkový kód pro operace s řetězci

Slova jsou textovými řetězci, ale čísla a data mohou také být textové řetězce, záleží na tom, jak je chcete použít a zobrazit. Z tohoto důvodu tato kapitola popíše oblasti využití, úlohy a operace vztahující se k textovým řetězcům.

Základní příkazy. V předchozích kapitolách jste se již naučili, jak například vynásobit telefonní čísla. Ale pravděpodobně budete souhlasit, že aritmetické operace se k tomuto typu textových řetězců příliš nehodí. Mnohem užitečnější by bylo třeba doplnit lokální telefonní číslo mezinárodním předčíslím. K tomu však musíte znát základní příkazy pro operace s textovými řetězci. To se právě naučíte v této kapitole.

Nejprve se můžete podívat na deklaraci textového řetězce.

Deklarace textových řetězců

Z technického pohledu mají pole pro různé účely různé datové typy. V jazyce ABAP existují dva základní datové typy pro textové řetězce:

- Datový typ `c` pro znaková pole.
- Datový typ `n` pro číselná pole.

Typ c. Datový typ `c` se používá pro alfanumerické znaky. Minimální délka pole je v tomto případě 1, maximální délka pole je 65 535 znaků. Výchozí hodnotou je mezera. Řetězcová pole musíte, stejně jako jiná pole, nejprve deklarovat. Pole GENDER s délkou jednoho znaku můžete deklarovat pomocí příkazu `DATA` (plný zápis) následovně:

```
DATA gender(1) TYPE c.
```

Jak vidíte, za názvem pole následuje bez mezery délka pole v závorkách a jako operand datový typ. Protože datový typ `c` je generický datový typ, můžete toto pole deklarovat také zkráceným způsobem:

```
DATA gender.
```

Výchozí nastavení

U generických datových typů je chybějící údaj nahrazen výchozím nastavením. Pokud nespecifikujete datový typ, systém automaticky doplní datový typ `c`; jestliže explicitně neuvedete délku pole s datovým typem `c`, systém implicitně vloží výchozí délku 1. Díky tomuto principu jsou předchozí deklarace pole GENDER totožné.

Jestliže chcete deklarovat pole LAST_NAME s datovým typem `c` o délce 20 znaků, můžete jednoduše použít následující příkaz:

```
DATA last_name(20).
```

Typ n. Pole s datovým typem `n` mají zvláštní využití. Přestože jsou tato pole určena pro textové řetězce, tyto textové řetězce se musí skládat pouze z číselných znaků. Tyto textové řetězce jsou v polích zarovnány doprava. Jestliže je pole delší než textový řetězec, volná místa jsou zleva doplněna nulami. Mezi běžné příklady těchto řetězců patří čísla článků, čísla položek, úrovně hierarchie a délky polí a záznamů. Kdykoliv chcete použít čísla a nemáte v úmyslu s nimi provádět aritmetické operace, měli byste použít datový typ `n`. Z tohoto důvodu se těmto polím říká *číselná textová pole*.

Datový typ `n` je rovněž generický datový typ. Minimální délka pole je 1 znak a maximální délka pole je 65 536 znaků, stejně jako u datového typu `c`. Oproti polím s datovým typem `c` mají však pole s datovým typem `n` výchozí hodnotu `0`.

LIKE. Další dodatky příkazu DATA fungují u obou datových typů stejně, zejména dodatky `LIKE` a `VALUE`. Například byste mohli deklarovat pole LAST_NAME stejným způsobem jako pole MNAME tabulky ZMEMBER01:

```
DATA last_name LIKE zmember01-mname.
```

Pole LAST_NAME bude tudíž mít stejné atributy jako pole MNAME – datový typ `c` a délku pole 20 znaků.

VALUE. Následuje příkaz pro přiřazení hodnoty „M“ poli GENDER:

```
DATA gender VALUE 'M'.
```

Malá/velká písmena

Výchozí hodnota je v předchozím příkazu zapsaná jako literál. V této chvíli je velmi důležité, jestli zapíšeme znaky malými nebo velkými písmeny. Tento rozdíl je zásadní především z hlediska možných budoucích dotazů, protože pokud hledáte velké písmeno „M“ v poli, které obsahuje malé písmeno „m,“ systém nic nenajde.

Operace s řetězci

Operace s řetězci. Mezi běžné operace s řetězci patří hledání skupiny znaků v řetězci, nahrazení znaku jiným znakem nebo posun znaků v řetězci doleva nebo doprava. Můžete dokonce chtít doplnit textový řetězec o znaky z jiného řetězce nebo odstranit z řetězce určité znaky.

Příklad

Podívejte se na jednoduchý příklad. Pokud chcete deklarovat telefonní číslo jako číselný textový řetězec s 9 znaky a přiřadit mu výchozí hodnotu „887766,“ odpovídající příkaz by vypadal takto:

```
DATA phone(9) TYPE n VALUE '887766'.
```

Dosud je vše jasné. Obsadili jste 6 míst, 3 místa jsou ještě k dispozici pro případ, že byste chtěli zpracovat delší telefonní číslo. Vypadá to, jako byste udělali vše správně, jelikož s telefonními čísly běžně neprovádíte aritmetické operace. Avšak v tomto příkladu jste zatím neuvažovali s tím, že výchozí hodnotou devítimístného číselného textového řetězce je devět nul. Vaše výchozí hodnota „887766“ přepíše šest nul zprava, takže na levé straně zůstanou tři nuly, obsahem daného pole tedy ve skutečnosti bude `000887766`.

Kdybyste zkusili vytočit toto telefonní číslo, pravděpodobně neuspějete. Musíte nějakou operací odstranit počáteční nuly z textového řetězce.

Posun textových řetězců

Jedním způsobem je posouvat textový řetězec doleva, dokud počáteční nuly nezmizí. To lze provést následujícím příkazem:

SHIFT.

```
SHIFT phone LEFT DELETING LEADING '0'.
```

Tento příkaz `SHIFT` posouvá obsah pole, textový řetězec „000887766“, doleva, dokud nesmaže všechny počáteční nuly. Po provedení operace bude pole obsahovat hodnotu `"887766 "`, to znamená, že za číslice se vpravo vložily tři mezery.

Textový řetězec můžete samozřejmě v poli posouvat libovolným směrem o zadaný počet míst. Pokud například chcete posunout telefonní číslo o tři místa doprava, použijete následující příkaz:

```
SHIFT phone BY 3 PLACES RIGHT.
```

Rozdíly mezi datovým typem c a datovým typem n. Po provedení příkazu bude obsah pole vypadat tak, jak jste původně zamýšleli – `"  887766"`. V tomto okamžiku si pravděpodobně říkáte, proč jste hned od začátku nepoužili datový typ `c`. Pokud použijete datový typ `c`, systém přijímá všechny alfanumerické znaky včetně speciálních znaků. Jestliže použijete datový typ `n`, systém tyto znaky ignoruje a přijímá jen číslice. Když se celý řetězec skládá jen z písmen a speciálních znaků, poli bude nastavena výchozí hodnota, což usnadňuje programování kontrol, které jsou schopny detekovat a zpracovat nesprávná vstupní data.

Datové typy

Rozhodnutí , který datový typ je vhodnější, skutečně závisí na dané situaci. Jestliže se číslo článku skládá z alfanumerických znaků, typ `c` je správnou volbou, pokud se však číslo článku skládá jen z číslic a vy chcete hned ze začátku zabránit vkládání nesprávných hodnot, lepší bude použít typ `n`.

Pokud vynecháte výše uvedené dodatky příkazu `SHIFT`, získáte tento příkaz v jeho nejjednodušším tvaru:

```
SHIFT phone.
```

Stejně jako v knihovně. V tomto případě systém použije výchozí nastavení pro chybějící operandy. Výchozí směr posunu je doleva a počet pozic je 1. Jinými slovy – obsah pole bude posunut o jednu pozici doleva. Při posunování obsahu pole byste však měli být opatrní. Představte si celou situaci jako knihovnu – pokud posunete knihy příliš daleko na jednu stranu police, některé z nich spadnou dolů. Jestliže posunete obsah pole za levý nebo pravý okraj, některé znaky budou nenávratně ztraceny.

Nahrazení znaků

Stejná procedura pro různé úlohy. Bez ohledu na to, zda chcete převést desetinné číslo z amerického formátu na český, nebo chcete vytvořit šestnáctimístné číslo materiálu podle starého osmimístného, vždy musíte vyhledat a nahradit určité znaky v textovém řetězci. K tomuto účelu slouží příkaz `REPLACE`.

Nechť pole PHONE_INTERNATIONAL má datový typ c, délku 25 znaků a obsahuje následující data – " 887766". První tři znaky jsou mezery. Nyní budete chtít nahradit počáteční mezery obsahem pole AREA_CODE. K tomuto účelu použijte následující příkaz:

```
REPLACE '  ' WITH area_code
             INTO phone_international.
```

REPLACE. Příkaz REPLACE vyhledá a nahradí první výskyt vyhledávaného řetězce – dvě mezery. Protože vyhledávaný řetězec je v textovém řetězci na první pozici, předchozí zápis nezpůsobí žádný problém. Pokud je AREA_CODE pole typu n o délce 5 a obsahuje hodnotu 09876, po provedení dané operace bude pole PHONE_INTERNATIONAL obsahovat hodnotu "09876 887766". První dvě mezery byly nahrazeny kódem oblasti, třetí mezera byla zachována a nyní se nachází mezi kódem oblasti a místním telefonním číslem.

Příklad

Abyste lépe pochopili nahrazování znaků v textových řetězcích, podívejte se na další příklad. Předpokládejte, že pole PHONE_INTERNATIONAL obsahuje hodnotu "09876 887766" a vy budete chtít nahradit počáteční nulu mezinárodním kódem a nulou v závorkách. Například pro Švýcarsko by příslušný řetězec byl "+41-(0)". Odpovídající příkaz by vypadal následovně:

```
REPLACE '0' WITH '+41-(0)' INTO phone_international.
```

Po provedení předchozího příkazu by dané pole obsahovalo hodnotu "+41-(0)9876 887766". Kdybyste chtěli nahradit zbývající mezeru mezi kódem oblasti a místním telefonním číslem znaménkem mínus, pak byste použili následující příkaz:

```
REPLACE ' ' WITH '-' INTO phone_international.
```

Tento příkaz hledá mezeru, kterou najde na dvanácté pozici a nahradí ji znaménkem mínus. Novým obsahem pole by byla hodnota "+41-(0)9876-887766".

Zhušťování textových řetězců

Odstranění nadbytečných mezer. Dlouhá textová pole často obsahují mezery, které jsou zbytečné. Tyto mezery mohou vzniknout třeba operacemi s řetězci v dlouhých textových řetězcích. Je možné spojit podřetězce tak, že všechny nadbytečné mezery budou odstraněny. Případně můžete seskupit textové řetězce z více polí do jednoho pole, kde je zhustíte.

Nejprve se podívejte na první možnost. Předpokládejte, že pole PHONE_INTERNATIONAL obsahuje hodnotu "+41 -(0)9876 887766". Zde je mezera mezi mezinárodním kódem a kódem oblasti a dvě mezery mezi kódem oblasti a místním telefonním číslem. Celkově tento textový řetězec obsahuje tři mezery.

CONDENSE. Příkaz CONDENSE vám umožní spojit tři části textového řetězce v poli PHONE_INTERNATIONAL tak, že mezi mezinárodním kódem "+41", kódem oblasti "-(0)9876" a místním telefonním číslem "887766" bude vždy jen po jedné mezeře, bez ohledu na to, kolik mezer zde bylo předtím. Proto bude odstraněna pouze třetí mezera a po provedení této operace bude pole PHONE_INTERNATIONAL obsahovat hodnotu "+41 -(0)9876 887766".

```
CONDENSE phone_international.
```

NO-GAPS. Pokud si stále myslíte, že by zde neměly být žádné mezery, můžete použít dodatek `NO-GAPS`, a odstranit tak, všechny mezery v textovém řetězci. V tomto případě byste použili následující příkaz:

```
CONDENSE phone_international NO-GAPS.
```

Po provedení tohoto příkazu bude pole obsahovat hodnotu `"+41-(0)9876887766"`.

Spojování textových polí

Oddělená výstupní pole. Existuje však mnohem jednodušší způsob, jak umístit textová pole do výsledného pole v libovolném pořadí. Představte si, že na řádek seznamu chcete vypsat dvě pole – NAME a FIRST_NAME. Obě pole jsou typu `c` a mají délku 40 znaků. To znamená, že tato dvě pole zaplní 80 znaků řádku seznamu, přičemž spousta z těchto znaků budou mezery. Bylo by tedy užitečnější vypsat obsahy obou polí do samostatného výstupního pole a spojit textové řetězce takovým způsobem, aby mezi příjmením a jménem zbyla jen jedna mezera.

CONCATENATE. Tuto metodu můžete také aplikovat na telefonní číslo. Představte si, že budete chtít zkopírovat obsah pole INTERNATIONAL_AREA_CODE, AREA_CODE a PHONE v uvedeném pořadí do pole PHONE_INTERNATIONAL. K tomu můžete použít tento příkaz `CONCATENATE`:

```
CONCATENATE international_area_code
            area_code phone
            INTO
            phone_international.
```

K získání informace o zdrojových polích používá příkaz `CONCATENATE` pozici operandů, pořadí zdrojových polí tudíž určuje pořadí textových řetězců v cílovém poli. Jinými slovy – pole INTERNATIONAL_AREA_CODE je první operand, takže obsah tohoto pole bude tvořit první část textového řetězce v poli PHONE_INTERNATIONAL, podobně pak obsah pole AREA_CODE představuje druhou část řetězce v cílovém poli a obsah pole PHONE třetí část tohoto řetězce. Pokud zdrojová pole mají datový typ `c`, počáteční mezery budou zkopírovány, zatímco mezery na konci budou ignorovány.

Příklad

Nechť je obsahem zdrojového pole INTERNATIONAL_AREA_CODE hodnota `"+41"`, obsahem pole AREA_CODE hodnota `"09876"` a obsahem pole PHONE hodnota `"887766"`. Po provedení operace spojení bude cílové pole PHONE_INTERNATIONAL obsahovat hodnotu `"+4109876887766"`.

SEPARATED BY. Určitě uznáte, že čitelnost tohoto cílového pole by mohla být lepší, kdyby jednotlivé části telefonního čísla byly odděleny znaménkem mínus. Za tímto účelem nabízí příkaz `CONCATENATE` dodatek `SEPARATED BY`. Zde musíte specifikovat proměnnou nebo literál, který bude vložen mezi jednotlivé části textového řetězce.

U tohoto příkladu můžete použít literál `'-'` a napsat:

```
CONCENATE internationa_area_code
          area_code phone
```

```
        INTO
        phone_international
        SEPARATED BY '-'.
```

Pro lepší čitelnost je předchozí příkaz rozepsán na více řádků, po jeho provedení bude pole PHONE_INTERNATIONAL obsahovat hodnotu `"+41-09876-887766"`.

Rozdělení textových řetězců

Obdobně byste si jistě dokázali představit opačný princip k příkazu `CONCATENATE` – rozdělení velkého textového řetězce na více částí. V předchozím příkladě byl například mezinárodní kód oblasti součástí většího textového řetězce. Bylo by však užitečné, kdybyste mohli rozdělit textový řetězec na mezinárodní kód oblasti, místní kód oblasti a místní telefonní číslo.

SPLIT. Tento problém můžete vyřešit příkazem `SPLIT`, který rozdělí textový řetězec do více polí.

```
SPLIT phone_international AT '-'
      INTO
      internationa_area_code area_code phone.
```

AT. Předchozí příkaz rozdělí obsah pole PHONE_INTERNATIONAL v místě oddělovače. Oddělovač je uveden za dodatkem `AT` a může jím být obsah proměnné nebo literál (jako v tomto příkladu). Cílová pole jsou opět zapsána jako operandy, kde pozice operandů určují výsledné uspořádání. Kromě toho se musíte ujistit, že cílová pole jsou dostatečně velká, aby uložila jednotlivé části řetězce.

Pokud pole PHONE_INTERNATIONAL před spuštěním předchozího příkazu obsahovalo hodnotu „+41-09876-887766“, pak po provedení této operace bude pole INTERNATIONAL_AREA_CODE obsahovat hodnotu `"+41"`, pole AREA_CODE hodnotu `"09876"` a pole PHONE hodnotu `"887766"`.

Operace s řetězci s přímým umístěním

Operace s řetězci jsou nezbytné i u další skupiny úloh. Někdy chcete přidat určitý počet znaků na dané místo v řetězci nebo naopak odstranit určený počet znaků od dané pozice v řetězci.

Přímé umístění. Předpokládejte, že pole PHONE_INTERNATIONAL obsahuje řetězec `"+41-(0)9876-887766"` a vy víte, že pokud v něm je obsažen mezinárodní kód oblasti, je vždy umístěn na prvních třech místech. K těmto třem znakům se můžete dostat přes přímé umístění. Chcete-li tedy vybranou část řetězce umístit do cílového pole INTERNATIONAL_AREA_CODE, pak použijte následující příkaz, obsahující zadání délky: :

```
international_area_code = phone_international(3).
```

Jednoduše řečeno – tento příkaz provede to, že jako nový obsah cílového pole INTERNATIONAL_AREA_CODE, uloží obsah zdrojového pole PHONE_INTERNATIONAL, a to od pozice 0, o délce tři znaky. Délku specifikujete v závorkách přímo za zdrojovým polem, mezi těmito závorkami a názvem zdrojového pole nesmí být žádné mezery.

Od středu. Často se však hledaný podřetězec nachází někde uprostřed celého textového řetězce. Použití předchozí metody tedy vyžaduje znalost výchozí pozice a délky podřetězce. Pokud se podíváte na předchozí příklad, můžete se třeba pokusit zjistit místní kód oblasti a uložit jej do samostatného pole. Musíte jen vědět, že místní kód oblasti vždy začíná za čtvrtým znakem a že se skládá ze sedmi znaků. Proto musíte zkopírovat znaky na pozicích 5, 6, 7, 8, 9, 10 a 11, což provedete tímto příkazem:

```
area_code = phone_international+4(7).
```

Typ c versus typ n

Na tomto příkladě lze krásně ukázat rozdíl mezi datovými typy `c` a `n`. Jestliže je cílové pole typu `c`, bude po provedení předchozí operace obsahovat hodnotu `"(0)9876"`. Pokud je však typu `n`, bude obsahovat hodnotu „09876“, protože systém u datového typu `n` ignoruje závorky, což jsou alfanumerické znaky.

Tímto způsobem můžete odstraňovat znaky z řetězce a manipulovat se znaky uvnitř řetězců. Za předpokladu, že budete chtít změnit mezinárodní kód oblasti z hodnoty `"41"` na `"33"`, použijete následující příkaz:

```
phone_international+1(2) = '33'.
```

Protože nový mezinárodní kód oblasti má také obsahovat znaménko plus, postačí, když nahradíte znaky na druhé a třetí pozici v poli PHONE_INTERNATIONAL literálem `'33'`.

Ukázkový kód pro operace s řetězci

Výpis 5.1. Report Z_STRING_OPERATIONS

```
*&---------------------------------------------*
*& Report  Z_STRING_OPERATIONS                 *
*&                                             *
*&---------------------------------------------*
*&                                             *
*&                                             *
*&---------------------------------------------*

REPORT  z_string_operations                    .

* Definice tabulek
TABLES zmember01.

* Definice řetězců
DATA: gender VALUE 'W',
      last_name LIKE zmember01-mname,
      phone(8) TYPE n VALUE '887766',
      area_code(5) TYPE n VALUE '09876',
      international_area_code(4) VALUE '+41',
      phone_international(25).

* Kontrolní výstup zdrojového pole
WRITE / 'Původní obsah pole'.
WRITE:   30 phone,
         40 area_code,
         50 international_area_code,
```

```
27           60 phone_international. "zatím prázdný
28  ULINE.
29  * Příkaz SHIFT
30  WRITE / 'Příkaz SHIFT'.
31  SHIFT phone LEFT DELETING LEADING '0'.
32  WRITE 30 phone.
33  SHIFT phone BY 2 PLACES RIGHT.
34  WRITE /30 phone.
35  ULINE.
36  * Příkaz REPLACE
37  WRITE / 'Příkaz REPLACE'.
38  phone_international = '  887766'.
39  REPLACE ' ' WITH area_code
                INTO phone_international.
40  WRITE 60 phone_international.
41  REPLACE '0' WITH '+41-(0)'
                INTO phone_international.
42  WRITE /60 phone_international.
43  REPLACE ' ' WITH '-' INTO phone_international.
44  WRITE /60 phone_international.
45  ULINE.
46  * Příkaz CONDENSE
47  WRITE / 'Příkaz CONDENSE'.
48  phone_international = '+41 -(0)9876  887766'.
49  CONDENSE phone_international.
50  WRITE 60 phone_international.
51  CONDENSE phone_international NO-GAPS.
52  WRITE /60 phone_international.
53  ULINE.
54  * Příkaz CONCATENATE
55  WRITE / 'Příkaz CONCATENATE'.
56  phone = '887766'.
57  SHIFT phone LEFT DELETING LEADING '0'.
58  phone_international = space.
59  CONCATENATE international_area_code
                area_code phone
60              INTO
61              phone_international
62              SEPARATED BY '-'.
63  WRITE 60 phone_international.
64  ULINE.
65  * Příkaz SPLIT
66  WRITE / 'Příkaz SPLIT'.
67  international_area_code = space.
68  area_code = space.
69  phone = space.
70  SPLIT phone_international AT '-'
71        INTO
72          international_area_code
            area_code
            phone.
73  WRITE: 30 phone,
74         40 area_code,
75         50 international_area_code.
76  ULINE.
77  * Přímé umístění
78  WRITE / 'Přímé umístění'.
79  international_area_code = space.
80  international_area_code =
                 phone_international(3).
```

```
81  WRITE 50 international_area_code.
82  phone_international = '+41-(0)9876-887766'.
83  area_code = space.
84  area_code = phone_international+4(7).
85  WRITE 40 area_code.
86  phone_international+1(2) = '33'.
87  WRITE 60 phone_international.
```

Poznámky ke zdrojovému kódu

Řádky 15 až 20

Nejprve návrh. Když v praxi navrhujete pole a datové typy, musíte nejprve analyzovat, jaký datový typ nejlépe splní vaše požadavky. Chyby v návrhu se v pozdější fázi vývoje těžce opravují. Příklad z výpisu 5.1 předpokládá, že místní telefonní číslo a kód oblasti jsou složeny jen z číslic. Mezinárodní kód oblasti však může obsahovat znaménko plus a celé mezinárodní telefonní číslo může navíc pro lepší čitelnost obsahovat lomítka, závorky nebo znaménka mínus. Z tohoto důvodu musí mít mezinárodní kód oblasti a mezinárodní telefonní číslo datový typ `c`.

Řádky 23 až 28

Definice vzhledu. Tyto příkazy jsou nutné pro výstup seznamu. Kvůli přehlednosti zobrazujete obsah pole PHONE od pozice 30 na daném řádku, obsah pole AREA_CODE od pozice 40 a obsahy polí INTERNATIONAL_AREA_CODE a PHONE_INTERNATIONAL od pozic 50 a 60. Navíc jsou v seznamu použity vodorovné čáry, které zvýrazňují jednotlivé oblasti.

Řádek 31

Na tomto řádku posouváte textový řetězec v číselném poli doleva, abyste odstranili počáteční nuly. Samozřejmě tento problém by se dal vyřešit i jinými způsoby, ale jako jednoduchá ukázka toto řešení zcela postačuje.

Řádek 33

Zde posouváte obsah pole o dvě místa vpravo.

Řádek 38

Novým obsahem pole bude literál `'  887766'`. Dvě mezery na začátku jsou nutné pro následující příklad.

Řádek 39

Na tomto řádku nahrazujete první mezeru obsahem pole AREA_CODE.

Řádek 41

V tomto poli nahrazujete první výskyt znaku `'0'` směrem zleva literálem `'+41-(0)'`.

Řádek 43

V tomto poli nahrazujete první výskyt mezery (`' '`) znaménkem mínus (`'-'`).

Řádek 48

Jako součást tohoto cvičení přiřazujete poli PHONE_INTERNATIONAL hodnotu `'+41 -(0)9876 887766'`. Dané pole tedy obsahuje celkem tři mezery.

Řádek 49

Podřetězce jsou v poli spojeny takovým způsobem, že je mezi nimi vždy maximálně 1 mezera.

Řádek 51

Z pole jsou odstraněny i zbývající mezery.

Řádky 56 a 57

Zde je použita podobná metoda pro nastavení výchozí hodnoty pro následující cvičení.

Řádek 58

SPACE. Pole PHONE_INTERNATIONAL zaplníte mezerami. Tento příklad k tomu používá klíčové slovo `SPACE`. Stejného výsledku byste však dosáhli přiřazením literálu `' '`.

Řádky 59 až 62

Tento příkaz je rozložen do čtyř řádků. Zaplňuje prázdné pole PHONE_INTERNATIONAL obsahy polí INTERNATIONAL_AREA_CODE, AREA_CODE a PHONE v takovém pořadí, v jakém byla tato pole zapsána. Pro lepší čitelnost jsou jednotlivé podřetězce v cílovém poli odděleny znaménkem mínus.

Řádky 67 až 69

Pole, nezbytná pro další cvičení, jsou naplněna mezerami.

Řádky 70 až 72

V poli PHONE_INTERNATIONAL hledáte oddělovač `'-'`. Protože toto pole obsahuje dva výskyty tohoto oddělovače, vzniknou tři podřetězce. Tyto podřetězce se zapíší do cílových polí v takovém pořadí, v jakém jste je uvedli v daném příkazu.

Řádky 73 až 75

Do seznamu vypisujete pole, která dříve obsahovala pouze mezery a nyní obsahují jednotlivé řetězce, s ohledem na jejich pozici.

Řádek 80

Pole INTERNATIONAL_AREA_CODE vyplňujete prvními třemi znaky pole PHONE_INTERNATIONAL.

Řádek 84

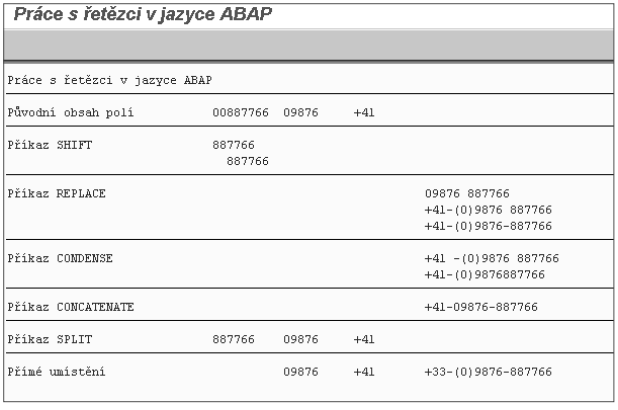
Systém zkopíruje sedm znaků pole PHONE_INTERNATIONAL za pozicí 4, jmenovitě znaky na pozicích 5, 6, 7, 8, 9, 10 a 11, a pokusí se je uložit do pole AREA_CODE. Zdrojové pole má datový typ `c` a obsahuje závorky na pozicích 5 a 7. Datový typ cílového pole je `n`, což znamená, že nepřijímá písmena a speciální znaky. Automatická konverze typů způsobí, že do cílového pole budou přeneseny jen číslice.

Řádek 86

V poli PHONE_INTERNATIONAL přepisujete znaky na pozicích 2 a 3 literálem `'33'`.

Výstup

Všechny kroky seznamu. Výstup seznamu začíná od pozice 30 původními hodnotami polí. To vám umožní ověřit si správnost seznamu. Pole PHONE obsahuje dvě počáteční nuly, zatímco pole PHONE_INTERNATIONAL neobsahuje vůbec žádnou hodnotu (viz obrázek 5.1). V druhé části byl daný textový řetězec nejprve posunut doleva a potom doprava. Třetí část obsahuje výsledky operací s příkazem `REPLACE`. Nejprve nahrazujete počáteční nuly mezinárodním kódem oblasti a poté mezeru znaménkem mínus. Ve čtvrté části jste použili příkaz `CONDENSE` pro odstranění mezer z textového řetězce a příkaz `CONCATENATE` z páté části vytvořil textový řetězec, ve kterém jsou dílčí řetězce odděleny znaménkem mínus. Dále pak příkaz `SPLIT` rozdělil řetězec do tří polí a poslední část ukazuje, jak lze použít přímé umístění pro změnu mezinárodního kódu oblasti z hodnoty `"41"` na `"33"`.



Obrázek 5.1. Obrazovka se seznamem, kterou získáte po spuštění kódu z výpisu 5.1